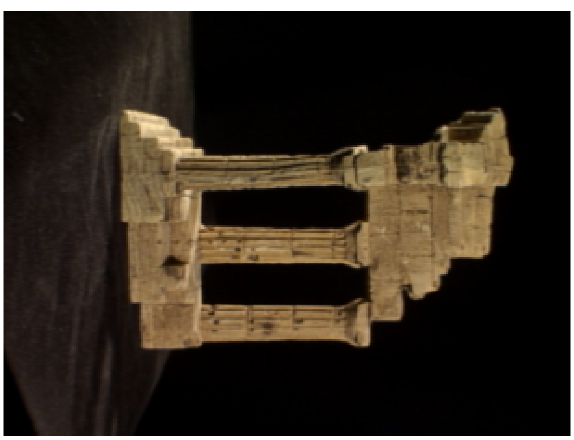
Name - **D.M.S.S.Dissanayake**
Index No - **190155L**

In [ ]:

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt

f = open(r'./templeSparseRing/templeSR_par.txt', 'r')
assert f is not None
n = int(f.readline())

# Reading the information on the first image
l = f.readline().split()
im1_fn = l[0]
K1 = np.array([float(i) for i in l[1:10]]).reshape((3,3))
R1 = np.array([float(i) for i in l[10:19]]).reshape((3,3))
t1 = np.array([float(i) for i in l[19:22]]).reshape((3,1))

# Reading the information on the second image
l = f.readline().split()
im2_fn = l[0]
K2 = np.array([float(i) for i in l[1:10]]).reshape((3,3))
R2 = np.array([float(i) for i in l[10:19]]).reshape((3,3))
t2 = np.array([float(i) for i in l[19:22]]).reshape((3,1))

# Read the two images and show
im1 = cv.imread(r'./templeSparseRing/' + im1_fn, cv.IMREAD_COLOR)
im2 = cv.imread(r'./templeSparseRing/' + im2_fn, cv.IMREAD_COLOR)
assert im1 is not None
assert im2 is not None

fig,ax = plt.subplots(1,2,figsize=(12,12))
ax[0].imshow(cv.cvtColor(im1,cv.COLOR_BGR2RGB))
ax[1].imshow(cv.cvtColor(im2,cv.COLOR_BGR2RGB))

for i in range(2):
    ax[i].axis("off")
```



In [ ]:

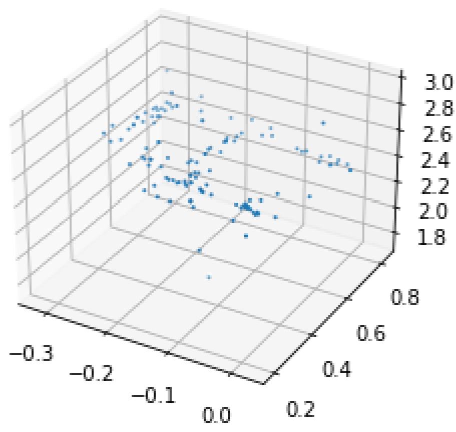
```
import numpy as np
import matplotlib.pyplot as plt

sift = cv.xfeatures2d.SIFT_create()
```

```
kp1, decs1 = sift.detectAndCompute(im1, None)
kp2, decs2 = sift.detectAndCompute(im2, None)

FLANN_INDEX_KDTREE = 1
index_params = dict(algorithm =FLANN_INDEX_KDTREE, trees = 5 )
search_params = dict(checks=100)
flann = cv.FlannBasedMatcher(index_params, search_params)
matches = flann.knnMatch(decs1, decs2, k=2)
good = []
pts1 = []
pts2 = []

for i, (m,n) in enumerate(matches):
    if m.distance < 0.7*n.distance:
        good.append(m)
        pts1.append(kp1[m.queryIdx].pt)
        pts2.append(kp2[m.trainIdx].pt)

pts1 = np.array(pts1)
pts2 = np.array(pts2)
F,mask = cv.findFundamentalMat(pts1, pts2, cv.FM_RANSAC) # Fundamental Matrix
print ("F:\n",F)
E = K2.T @ F @ K1 # Essential Matrix
print ("E:\n",E)
retval, R, t, mask = cv.recoverPose(E, pts1, pts2, K1)
R_t_1 = np.concatenate((R1, t1), axis =1) # 3 x 4
R2_ = R1 @ R
t2_ = R1 @ t
R_t_2 = np.concatenate((R2_, t2_), axis =1) # 3 x 4
P1 = K1 @ np.hstack((R1, t1))
P2_ = K2 @ R_t_2
```

```
F:
 [[ 1.19353197e-06  1.48128487e-05 -2.65668422e-02]
 [-8.37167541e-06  6.34793204e-07  2.04080864e-03]
 [ 2.41439516e-02 -5.73622910e-03  1.00000000e+00]]
E:
 [[ 2.75898779e+00  3.43654884e+01 -3.42837514e+01]
 [-1.94221058e+01  1.47803397e+00 -5.08742503e-01]
 [ 3.41148335e+01 -1.68046954e+00 -1.62748485e-02]]
```

In [ ]:

```
points4d = cv.triangulatePoints(P1, P2_, pts1.T, pts2.T)
points4d /= points4d[3, :]
import matplotlib.pyplot as plt
X = points4d[0, :]
Y = points4d[1, :]
Z = points4d[2, :]
fig = plt.figure(1)
ax = fig.add_subplot(111, projection='3d')
ax.scatter(X, Y, Z, s=1, cmap='gray')
plt.show()
```


3.0
2.8
2.6
2.4
2.2
2.0
1.8
0.8
0.6
0.4
0.2
−0.3
−0.2
−0.1
0.0